Managing and Installing Python Packages in Conda Environments

You have two options to install your own Python packages in our machine learning environment:

- Use the pip tool to install them directly
- Build your own conda environment

Consider the benefits and disadvantages of each method, described below, before choosing which works best for you.

Using the pip Tool to Install Packages

Using the pip tool to install packages is easier than building your own conda environment and takes less space in your home directory.

The packages installed with pip will be applied to all environments that you load and will take precedence over the installed versions of packages within those environments—possibly causing version errors, depending on the differences. If the packages do not show up in other environments, make sure the Python versions in the loaded environment and the base environment match. Packages are Python-version-dependent; therefore, to ensure you can use the packages with the environment you want, load the environment before using pip.

The packages will be installed in the $HOME/.local/lib/... directory.

Complete the following steps to install packages on a Pleiades front-end system (PFE):

1. Load the miniconda3 module:

   ```
   pfe20 % module use -a /swbuild/analytix/tools/modulefiles
   pfe20 % module load miniconda3/v4
   ```

2. Use the pip command to install the packages:

   ```
   pfe20 % pip install --user package_name
   ```

Building Your Own Conda Environment

Building your own conda environment gives you the control to manage and install your own packages, and they will be less likely to have version errors than the pip-installed packages. The easiest way to create your own environment is to clone an existing conda environment into your own directory, then modify it.

Creating an environment can take up a significant portion of your disk quota, depending on the packages installed. To ensure that you can use your conda environment properly, please familiarize yourself with all the basic conda commands.

Before You Begin: Install a conda Token

The conda environments are supported under an Anaconda site license. In order to create a conda environment and be in compliance with the NASA Ames site license, we require all users to install a conda token prior to creating an environment. Complete these steps to acquire and install a token:

1. Request a NASA Ames conda token by sending email from your NAS account to: dataanalytics@nas.nasa.gov. You will receive an email, “Invitation to the NASA Ames organization on Anaconda Nucleus.”
   Important: Email sent from a non-NASA email address will not receive a response.
2. Create an Anaconda Nucleus account using the email you received. Click the **Join Organization** button and complete creating an account using the email address that the invite was sent to.
3. In two or three days, you will receive an email from Anaconda Nucleus with your new token and instructions on how to install it.

To activate your token on a NAS system:

1. Log on to a PFE.
2. Load the HECC Data Science miniconda module as follows:

3. ```
   module -a use /swbuild/analytix/tools/modulefiles
   module load miniconda3/v4
   ```

4. Run the conda token set command with token you received in the email:
   conda token set *token_string_from_email*
5. Verify the token is linked by using the conda info command. In the channel URLs, there should be:
   https://repo.anaconda.cloud/repo/main/...
   instead of:
   https://repo.anaconda.com/pkgs/main/...
   In addition, you can check whether https://repo.anaconda.cloud/repo/main appears in your .condarc file.

You should now have access to the commercial anaconda.cloud repo. Environments cloned from one of the Data Science conda environments or created from one of the Data Science YAML files will download and install packages from this repo.

If you have questions about this procedure or require assistance, please contact us by sending email to support@nas.nasa.gov.
   ```

Creating a Conda Environment from a Clone

Cloning from a YAML File

To build your own conda environment based on one of the NAS-provided conda environments, you can use the YAML file for the environment you want to clone. A list of YAML files for each of the NAS-provided conda environments can be found in the /swbuild/analytix/tools/miniconda3_220407_yml/ directory. For example:

- dask_mpi.yml
- horovod_env.yml
- jupyterlab_env.yml
- pyt1_10_2_env.yml
- pyt1_11_env.yml
- pyt1_12_env.yml
- pyt1_13_env.yml
- pyt1_8_env.yml
- r3_6_env.yml
- smartg_env.yml
- tf1_15_env.yml
- tf2_10_env.yml
- tf2_8_env.yml
- tf2_9_env.yml

For more information about YAML files, see the conda documentation.

Complete these steps to build a conda environment from a YAML file:

1. Load the miniconda3 module:

   ```
   pfe20 % module use -a /swbuild/analytix/tools/modulefiles
   pfe20 % module load miniconda3/v4
   ```

   TIP: If you don't want to fill up your home directory, you can put the environment in your/nobackup directory. To do this, set the CONDA_ENVS_PATH environment variable to point to a .conda/envs folder in /nobackup, as shown below. The folder is created the first time you run the conda command.
   - For bash: % export CONDA_ENVS_PATH=/nobackup/$USER/.conda/envs
   - For csh: % setenv CONDA_ENVS_PATH /nobackup/$USER/.conda/envs
2. Export the path to the target directory where you want to put the packages for installation:

   ```
   pfe20 % export CONDA_PKGS_DIRS=/nobackup/$USER/.conda/pkgs
   ```

3. Copy the YAML file you want to use to your local directory:

   pfe20 % cp /swbuild/analytix/tools/miniconda3_220407_yml/*filename*.yml *filename*.yml.

4. Open the file, then change the entry for name to a name for your new environment.

   Note: This step is required because keeping the default name may cause conflicts in cloning the environment.

5. At the prompt, enter the conda create command to create the environment:

   pfe20 % conda env create -f *filename*.yml

Cloning with the conda clone command

If the environment you want to clone is small , or you want to clone an environment based on one that is not provided by NAS, you can use the conda clone command to copy an existing environment to your local directory.

Complete these steps to build your own conda environment by cloning an existing environment:

1. Load the miniconda3 module:

   ```
   pfe20 % module use -a /swbuild/analytix/tools/modulefiles
   pfe20 % module load miniconda3/v4
   ```

2. Export the path to the target directory where you want to put the packages for cloning:

   pfe20 % export CONDA_PKGS_DIRS=/nobackup/$USER/*target_directory*

3. Clone an existing conda environment into a new folder. In this example, we create a new environment with no extra packages installed by cloning the base environment:

   pfe20 % conda create --name *my_env* --clone base

Your environment has been created.

Creating a Fresh Conda Environment

Another option is to create a fresh conda environment. Complete these steps to do so:

1. Load the miniconda3 module:

   ```
   pfe20 % module use -a /swbuild/analytix/tools/modulefiles
   pfe20 % module load miniconda3/v4
   ```

2. Export the path to the target directory where you want to put the packages for installation:

   ```
   pfe20 % export CONDA_PKGS_DIRS=/nobackup/$USER/.conda/pkgs
   ```

3. At the prompt, enter the conda create command to create the environment:

   pfe20 % conda create -n *my_env* python=3.x

Using Your Custom Environment

To use your custom environment, follow these steps:

1. Load the miniconda3 module:

   ```
   pfe20% module use -a /swbuild/analytix/tools/modulefiles
   pfe20% module load miniconda3/v4
   ```

2. Activate your environment:

   (for bash)
   pfe20% source activate *my_env*

   (for csh)
   pfe20% source /swbuild/analytix/tools/miniconda3_220407/bin/activate.csh *my_env*

   Note: If you created your environment in your /nobackup directory, use:

   (for bash)
   pfe20% export CONDA_ENVS_PATH=/nobackup/$USER/.conda/envs
   pfe20% source activate *my_env*

   (for csh)
   pfe20% setenv CONDA_ENVS_PATH /nobackup/$USER/.conda/envs
   pfe20% source /swbuild/analytix/tools/miniconda3_220407/bin/activate.csh *my_env*

3. Install any other packages you want by using the conda install command:

   *(my_env)*pfe20% conda install *package_name package_name* ...

Your newly installed packages are ready to use. After you finish using your environment, you can deactivate it by using theconda deactivate command.

If you have any issues, please contact us at support@nas.nasa.gov.

For more information about NAS-provided conda environments, see Using Conda Environments for Machine Learning.

Article ID: 627
Last updated: 11 Oct, 2023
Updated by: Moyer M.
Revision: 76
Machine Learning -> Machine Learning at NAS -> Managing and Installing Python Packages in Conda Environments
https://www.nas.nasa.gov/hecc/support/kb/entry/627/